**MFG 598: - Engineering Computing with Python**

**Fraud Detection in Credit Card Transactions**

Kaushik Shankar (1225653286)

# **Table of contents**

## **Introduction**

## **Data Analysis**

## **Pre-processing Dataset**

## **Machine Learning Model Implementation**

### **K Nearest Neighbours**

### **Random Forest**

### **Decision tree**

### **Logistic Regression**

### **Support Vector Machine**

**Results**

**References**

# **Introduction**

The Problem involves the detection of credit card fraud. It is important to be able to detect fraud to stop it and this process is to be done as fast as possible. Given the nature of the problem it is very difficult to be detected by a human, to go over the data and understand the trends in the data to detect any anomalies. Also, human labor for such a problem will require a lot of time which is not ideal. So, in order to solve this problem one of the best methods to use is machine learning. During the project the team accessed the dataset and tested which machine learning was most efficient in giving proper results. It was an opportunity for the team to implement and understand the methods that were discussed in the class in depth.

# **Data Analysis**

The dataset was collected from Kaggle. Upon statistical analysis of the dataset, we observed that the dataset was heavily imbalanced in the classes. The dataset has 492 fraud transactions out of 284,807 transactions, that is 0.172% of fraud transactions in the entire dataset (fig.1.). Also, the dataset consists of 28 features which are numerical information from a PCA transformation for confidentiality reasons. There are two more columns that are ‘Time’ and ‘Amount’, considering that time has no proper causality we decided to drop this feature from the dataset.

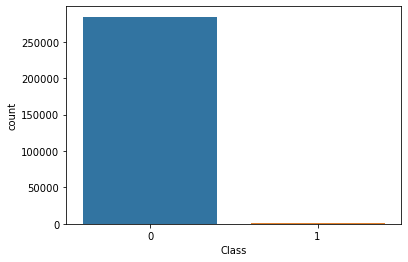


Fig.1. Count of Class 0 and Class 1

Performing univariate analysis (fig.2.) showed the features correlation with the ‘Class’. Bivariate analysis showed that the feature correlation among each other was very low. We decided to not drop any features as any small information could be crucial for a model in the process of prediction.

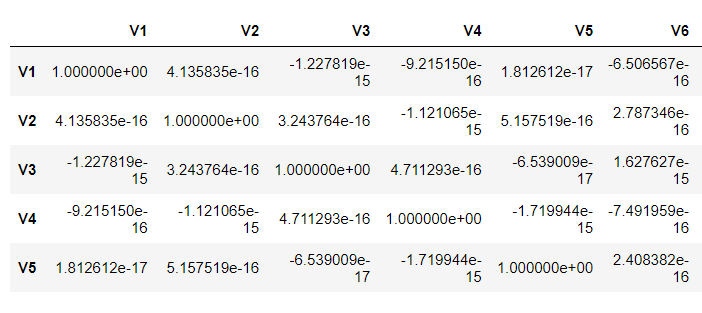
Table

Description automatically generated

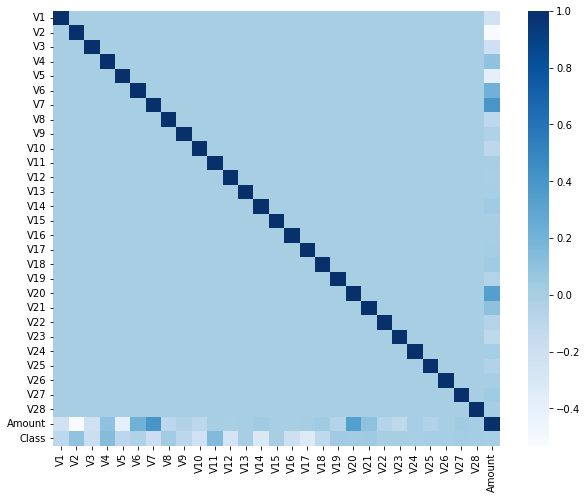
The fig.2. Besides shows that the feature V17 is highly correlated with class and feature V22 correlation with class is very low.

This analysis helped in the further steps of the project, to decide from which features the outlier points can be removed such that it does not have a higher impact on the prediction of Class.

Fig.2. Univariate heatmap of features with class

Fig.3. Correlation matrix values

The correlation matrix of the dataset consists of 30x30 elements, some of the values are shown above in fig.3. and followed with a heatmap which allows for easier visualization of such data.



The heatmap fig.4. Reflects the same information as we see above that correlation between the features is very low.

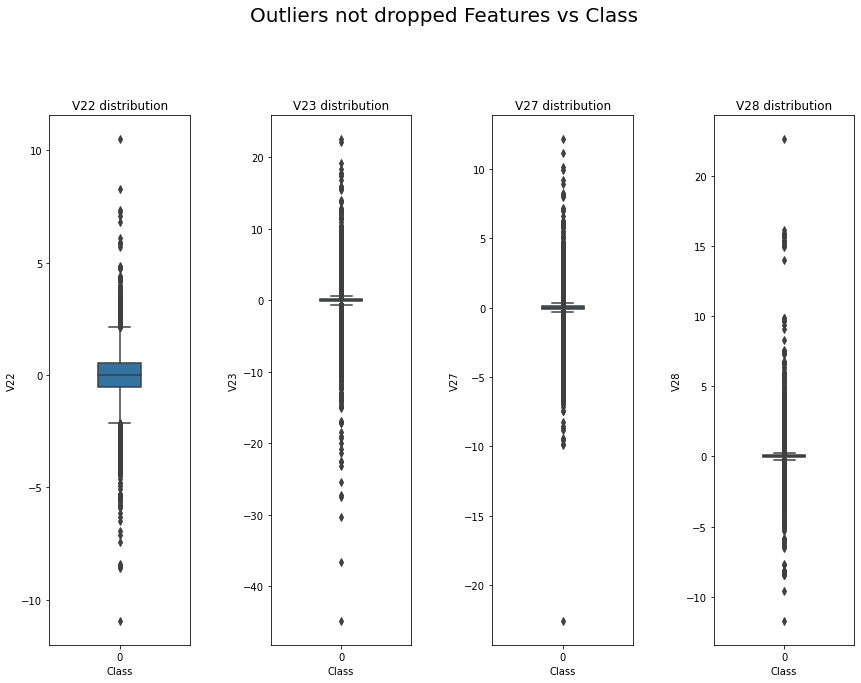
The red box highlights that all the pixels of the heatmap have a similar color to that near to value 0.0.

# 

# 

Fig.4. Correlation matrix heatmap

In this dataset there are a lot of outliers fig.5., it can be seen in the boxplot done for some of the features for Class 0 (as an example). Outliers in a dataset are not always a bad thing, these points still can contain information which can be essential for the prediction process.



# 

Fig.5. Boxplot of features V22, V23, V27, V28.

# 

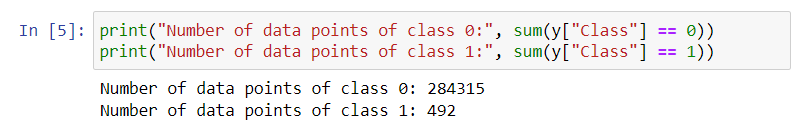
# 

# 

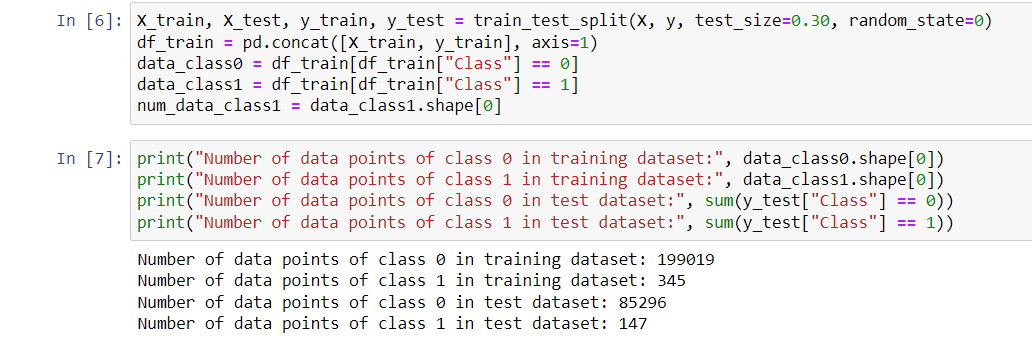
# **Pre-processing Dataset**

**Dataset Split**

The entire dataset consists of 284315 Class 0 data points and 492 Class 1 data points.



Among these points when we split the dataset into train and test dataset, with the test dataset being 30 % that of the entire dataset, we got 199019 Class 0 and 345 Class 1 in the train dataset and the rest in the test dataset.



**Downsampling Dataset**

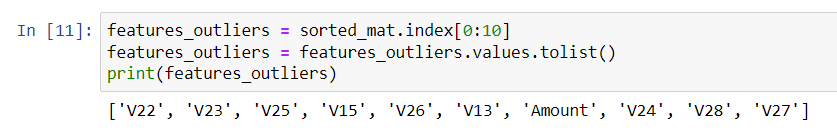
As we have seen earlier the dataset is extremely imbalanced. To handle such a dataset there are many methods, the technique we chose for this project was to downsample the majority Class i.e. Class 0 and train the models with the downsampled dataset.

The downsampling of the dataset was done in two different ways.

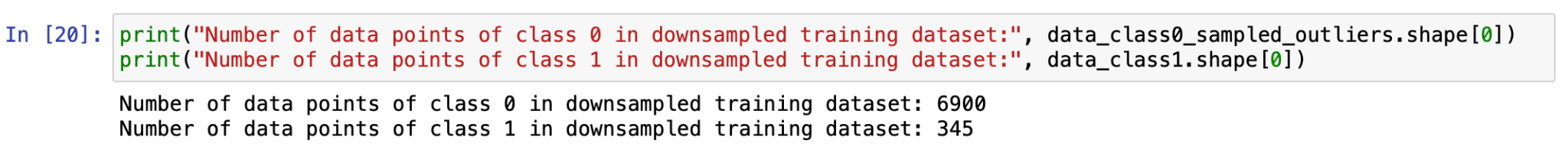
*First method*, we did not remove any data points from the train dataset we got from the test train split. We downsampled the Class 0 data points randomly from that training dataset.

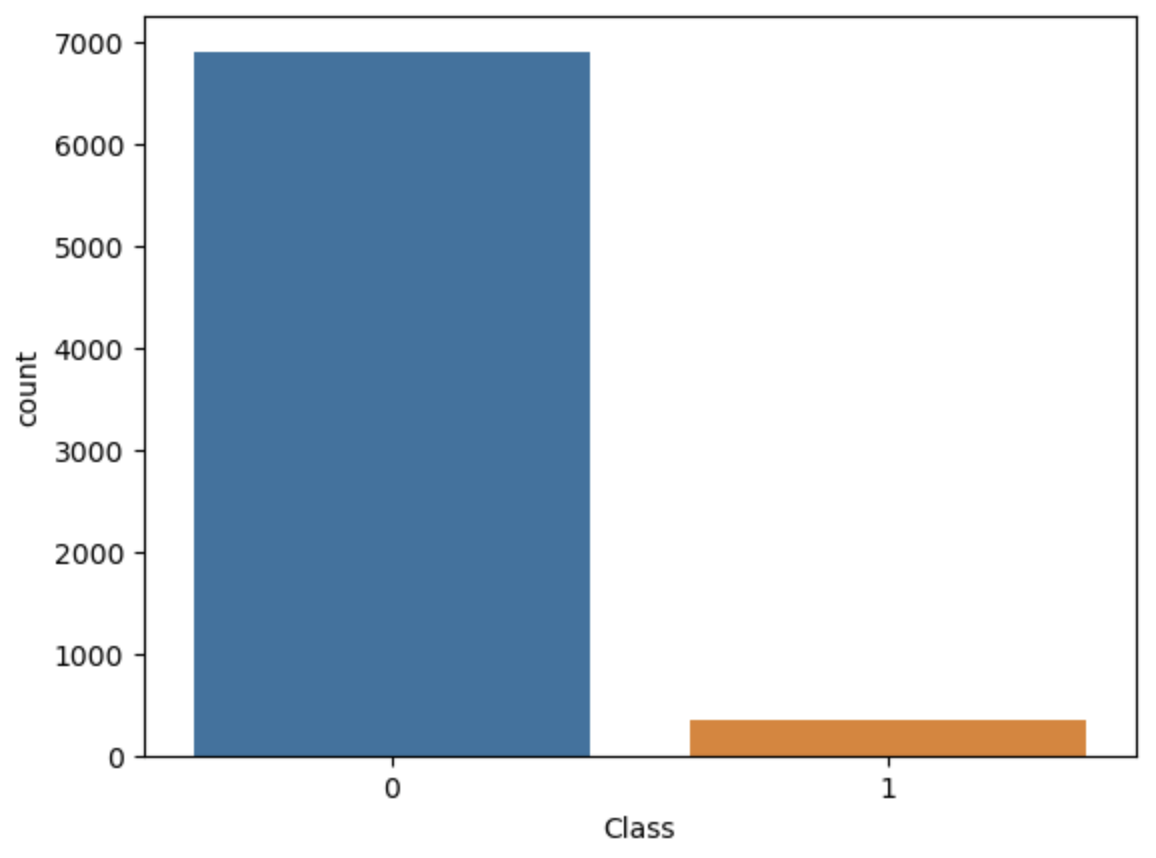
*Second method*, we removed the outlier points from the train dataset of Class 0 of features that had very low correlation with the target ‘Class’. The Features considered for removal of outliers were: ['V22', 'V23', 'V25', 'V15', 'V26', 'V13', 'Amount', 'V24', 'V28', 'V27'].

We decided not to remove any outlier’s data points of Class 1 as it is the minority class.



For both the methods the number of data points of Class 0 were sampled such that they were 20 times greater than Class 1 data points. So, the final two downsampled dataset consist of 6900 Class 0 data points and 345 Class 1 data points.





Beside is the count plot, fig.6. of the one of the trainings dataset we used for all the Machine learning models.

Fig.6. Count of Class 0 and Class 1 of downsampled dataset

**Process used to detect outliers.**

The method we used to detect the outliers was Interquartile Range of Detect. We Decided to remove outliers because most of the machine learning algorithms are negatively impacted by the presence of outliers.

Below Image shows where the outliers lie, fig.7.

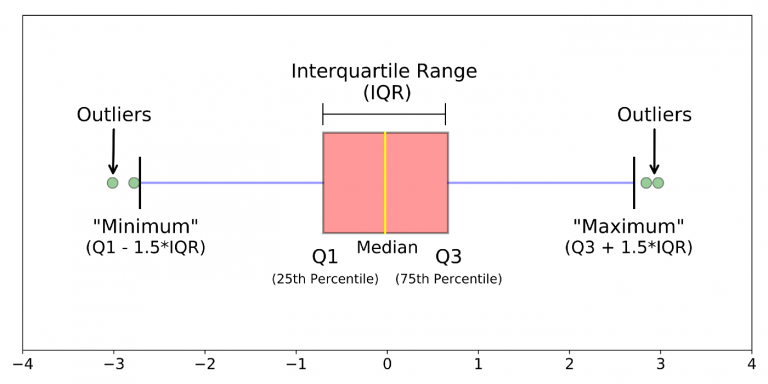
[](https://towardsdatascience.com/practical-guide-to-outlier-detection-methods-6b9f947a161e)

Fig.7. Visualization of outlier points

The dataset is divided into 4 equal parts, where we can see from the image that

Q1 = 25th percentile

Q2 = median

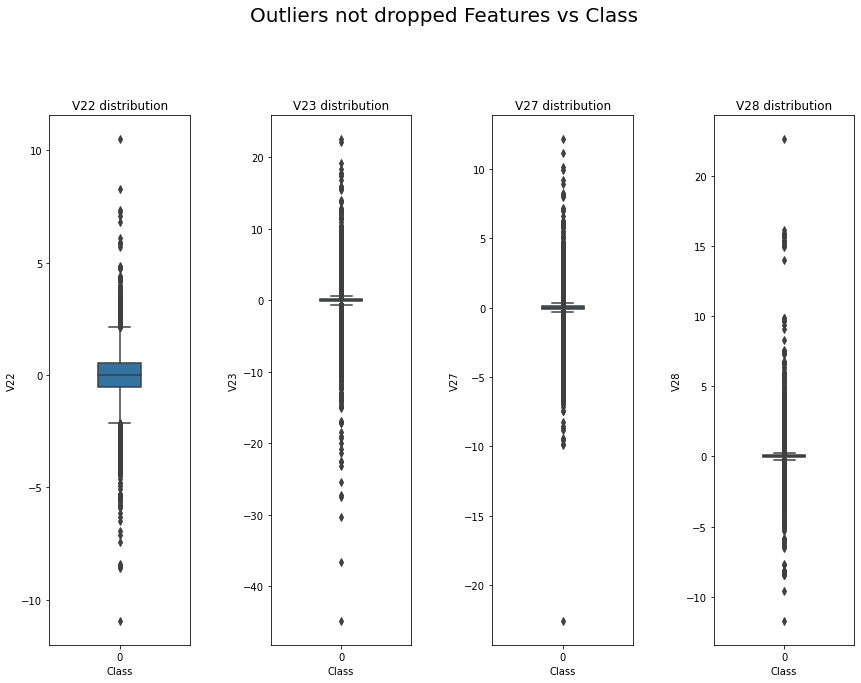
Q3 = 75th percentile

IOR = Q3 - Q1.

Any point that lies below Q1 - IQR\*1.5 and above Q3 + IQR\*1.5 are considered outliers.

In a normal distribution these points would be at the tail of the distribution.

In our dataset we can see the comparison of the features distribution in a box plot before and after removal of the outlier points of some features (V22, V23, V27, V28).



The boxplot is made for the features ‘V22, V23, V27, V28’ as an example.

We can see that the dataset consists of a lot of outliers, fig.8. and upon removing the outliers we can see the change in the boxplot in fig.9.

Fig.8. Boxplot features V22, V23, V27, V28 before removing outliers.

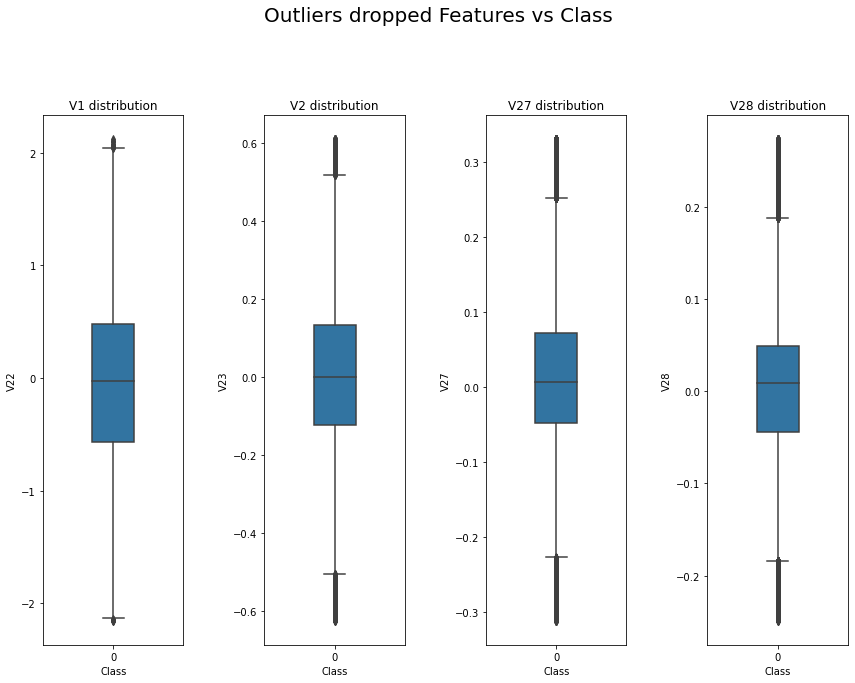


Fig.9. Boxplot of features V22, V23, V27, V28 after removing outliers.

**Normalization of the Dataset**

It is important to Standardize the dataset so that the features are all on a similar scale and it helps with all the variables having equal weight on the model. This improves the stability of the Machine learning model.

We used the Robust Scaler method to normalize the dataset because this method is resistant to outliers and as the dataset consists of a lot of outliers it was important to not let those points affect that dataset. The Robust scaler method works similar to IOR method, it does not include the points below Q1 - IQR\*1.5 and above Q3 + IQR\*1.5 and the points within the range standardized using standard scalers.

# **Machine Learning Model Implementation**

Three datasets were considered for the training of the machine learning models.

* Entire training dataset (199019 Class 0, 345 Class 1)
* Downsampled training dataset consisting of outliers (6900 Class 0, 345 Class 1)
* Downsampled training dataset not consisting of outliers (6900 Class 0, 345 Class 1)

Machine learning models trained.

* K Nearest Neighbors
* Random Forest
* Decision Tree
* Logistic Regression
* Support Vector Machine

**K Nearest Neighbors**

K Nearest Neighbors is one of the supervised learning methods. It is a non-parametric method. KNN saves the training data instead of learning from it, and when it receives a new data point, it categorizes it based on how well it matches with pre-existing categories.

Hyper parameters chosen: We used Sklearn library to implement the KNN algorithm. In the library we are allowed to choose the number of neighbors that influence the algorithm in categorizing the new data point. We chose the n\_neighbors: 6, this was chosen by looping over a range of values and this value gave a higher AUC score. We chose the weights: ‘distance’, this implies the closer neighbors of a new data point will have greater influence than the neighbors that are further away.

**Random Forest**

Random forest is a supervised machine learning algorithm, a general purpose classification and a regression method. Random forests or random decision forests is an ensemble, which combines the output of multiple decision trees. Based on the predictions made by the decision trees, this algorithm determines the outcome. The accuracy of the result grows as the number of trees increases.

Hyper parameters chosen: We used Sklearn library to implement the Random Forest algorithm. In the library, n\_estimators are the number of decision trees in the forest. By observation, we chose the n\_estimators: 30 which gave a higher AUC score.

**Support Vector Machine**

An algorithm for supervised machine learning is the support vector machine (SVM). It can be used for both classification and regression. Finding a hyperplane in an N-dimensional space is the primary objective of the SVM method. The data points from the dataset are precisely classified by this hyperplane. The hyperplane's dimension is determined by the number of features. When there are just two input characteristics, the hyperplane is essentially a line. The hyperplane turns into a 2-D plane if the number of input characteristics is three. We find it a little challenging to imagine an output with more than three features.

Hyperparameters Chosen: We have used the Sklearn library for the implementation of the SVM algorithm. We used the max\_iter to set a hard limit on the number of iterations, regularization parameter, gamma which determines how far the influence of a single training example reaches and we use the default ‘rbf’ type of kernel. The random\_state is taken as an integer and the function produces the same result across different executions.

**Logistic Regression**

Logistic Regression is a supervised machine learning algorithm. Classification issues are resolved via logistic regression. In contrast to linear regression, which predicts a continuous outcome, it achieves this by forecasting categorical outcomes. Logistic regression works with binary data, where either the event happens (1) or the event does not happen (0). So given some feature x it tries to find out whether some event y happens or not. So, y can either be 0 or 1. In the case where the event happens, y is given the value 1. If the event does not happen, then y is given the value of 0.

Hyperparameters: We have used the Sklearn library for the implementation of logistic regression algorithms. In the library, we have provided the x-training set and y-training set as input to learn the relationship between x and y training set. A new machine learning model is developed with the help of these 2-training dataset and logistic regression algorithms. Now, at the end a new x-test dataset is given as input to the ML model to predict the outcome.

**Decision Tree**

A decision tree is a flowchart-like structure in which each leaf node represents a class label, each internal node represents a "test" on an attribute and each branch indicates the result of the test. A non-parametric supervised learning technique for classification and regression is decision trees. The objective is to learn straightforward decision rules derived from the data features to build a model that predicts the value of a target variable. It is easy to comprehend and interpret. One can picture a tree. able to manage issues with several outputs. Overly complicated trees that poorly generalize the input can be produced by decision tree learners. Overfitting is the term for this. To circumvent this issue, mechanisms like pruning, establishing the minimum number of samples needed at a leaf node, or establishing the maximum depth of the tree are required.

We will use the Decision Tree Classifier class from the Sklearn library to train and evaluate models. We use X\_train and y\_train data for training purposes. X\_train is a training dataset with features, and y\_train is the target label.

**Results**

The metrics chosen to evaluate the models were AUC score, True Positive Rate (TPR), False Positive Rate (FPR) and ROC Curve.

Below is the table fig.10. consisting of the machine learning models and the dataset used to train those models.

Graphical user interface, table

Description automatically generated

Fig.10. Results of models

From the results table we can see that the AUC score and TPR is more when the models are trained with a downsampled dataset not consisting of outliers, but we can observe the FPR decreases. There is an increase in the FPR since the dataset is extremely imbalanced and the number of miss predictions are high in Class 0.

Observing the results, dataset trained with a downsampled dataset consisting of outliers have a good balance of TPR and FPR values. We observed that with respect to AUC Score Logistic Regression model performed well. Also, Random Forest has a good balance of TPR and FPR values. Random Forest model trained on entire train dataset compared to trained on downsampled dataset consisting outliers the value TPR increased from 0.74 to 0.83 and FPR is at 0.99

**ROC Curve**

Chart

Description automatically generated

The curve beside is for the models trained on complete train dataset.

The ROC curve also shows that the decision tree is not a good model and logistic regression is the best model among the five.

**Chart

Description automatically generated with medium confidence**

The curve beside is for the models trained on downsampled dataset with outliers.

The ROC curve also shows that the decision tree is not a good model and logistic regression, random forests have close curves, and these are the best model among the five.

**Chart, line chart

Description automatically generated**

The curve beside is for the models trained on a downsampled dataset with no outliers.

The ROC curve also shows that the decision tree is not a good model and logistic regression is the best model among the five.

Comparison of ROC curves of models trained on three datasets.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  | |

**Conclusion**

* Among all the models, Logistic Regression seems to be the best model to find the fraud cases occurring while processing credit card transactions.
* From the obtained results we were able to conclude that dataset consists of information in outlier points,
* Accuracy is not a good metric to evaluate models that are trained on an extremely imbalanced dataset.
* Using ensemble methods with proper weights will help with the prediction of Classes better because models trained on a downsampled dataset can classify Class 1 better and models trained on a complete dataset can classify Class 0 better. So, using an ensemble method will give better classification.

**References**

* <https://arxiv.org/pdf/1904.10604.pdf>
* <https://www.kaggle.com/datasets/whenamancodes/fraud-detection>
* <https://developers.google.com/machine-learning/data-prep/construct/sampling-splitting/imbalanced-data>
* <https://machinelearningmastery.com/robust-scaler-transforms-for-machine-learning/>
* https://towardsdatascience.com/practical-guide-to-outlier-detection-methods-6b9f947a161e
* <https://towardsdatascience.com/why-1-5-in-iqr-method-of-outlier-detection-5d07fdc82097>
* https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html